**Timestamps in Locking Protocols**

- **Timestamps:**
  - used to avoid deadlock.
  - each transaction has a single timestamp.
  - timestamps are used to resolve conflicts between transactions.

- **Possible Actions:**
  - wait: defer until conflicting transaction completes/aborts
  - restart: die - begin again but with original timestamp
  - wound - attempt to cause the conflicting transaction to die and continue when the conflicting transaction completes/aborts

- **Two algorithms:**
  - wait-die: non-preemptive; a transaction finding a conflict waits if it is older and dies if it is younger
  - wound-wait: preemptive; a transaction finding a conflict wounds if it is older and waits if it is younger

---

**Basic Timestamp Ordering (BTO)**

- **Read Operation:**
  - `read <object, TS>`
  - if `TS < W-ts`
    - reject/abort
  - else `R-ts = max(R-ts, TS)`

- **Write Operation:**
  - `write <object, val, TS>`
  - if `TS < R-ts or TS < W-ts`
    - reject/abort
  - else `W-ts = TS`

Thomas Write Rule: do not abort conflicting writes, simply ignore them.

---

**Multiversion Timestamp Ordering**

Read Operation:

- `read <object, TS>`
  - if (non-empty `WQ`) and `TS < W-vs`
    - reject/abort
  - else add the read operation to `RQ`

Write Operation:

- `write <object, val, TS>`
  - if (non-empty `RQ`) and non-empty `WQ` and `TS < R-vs`
    - execute the write operation
  - else add the write operation to `WQ`

---

**Conservative Timestamp Ordering**

- Each Data Manager maintains:
  - a read queue (`RQ_i`)
  - a write queue (`WQ_i`)

For each Transaction Manager, `TM_j`:

- `TS(Q_j)` denote the timestamp of the first operation in `Q_j`

Scheduler:

- `DM_k` (other Data Managers)

---

**Optimistic Algorithms (Kung-Robinson)**

- Each transaction, `T`, has three phases:
  - read phase
    - read from database and write to temporary storage (log)
  - validation phase
    - If (T does not conflict with any other executing transaction)
      - assign the transaction a unique (monotonically increasing) sequence number and perform the write phase
      - else abort T
  - write phase
    - write log to database
Optimistic Algorithms (Kung-Robinson)

Let:
- $t_s$ be the highest sequence number at the start of $T$
- $t_f$ be the highest sequence number at the beginning of $T$'s validation phase

validation algorithm:

valid = true;
for $t = t_s + 1$ to $t_f$, do
  if ($\text{writeset}[t] \cap \text{readset}[T] \neq \emptyset$)
    then valid = false;
  if (valid)
    then do write phase;
        increment counter;
        assign $T$ a sequence number;